Day 1: Overview of Core Concepts

ME414: Introduction to Data Science and Big Data Analytics

LSE Methods Summer Programme

14 August 2017

Outline

Overview of Core Concepts

Philosophy

- It is important to understand the ideas behind the various techniques, in order to know how and when to use them.
- One has to understand the simpler methods first, in order to grasp the more sophisticated ones.
- It is important to accurately assess the performance of a method, to know how well or how badly it is working (simpler methods often perform as well as fancier ones!).
- This is an exciting research area, having important applications in science, industry and policy.
- Machine learning is a fundamental ingredient in the training of a modern data scientist.

Two main approaches to machine learning

- Supervised Learning
- Unsupervised Learning

The Supervised Learning Problem

Starting point:

- Outcome measurement Y (also called dependent variable, response, target).
- Vector of p predictor measurements X (also called inputs, regressors, covariates, features, independent variables).
- In the regression problem, Y is quantitative (e.g price, blood pressure).
- In the classification problem, Y takes values in a finite, unordered set (survived/died, digit 0-9, cancer class of tissue sample).
- We have training data $(x_1, y_1), \ldots, (x_N, y_N)$. These are observations (examples, instances) of these measurements.

Objectives

On the basis of the training data we would like to:

- Accurately predict unseen test cases.
- Understand which inputs affect the outcome, and how.
- Assess the quality of our predictions and inferences.

Unsupervised Learning

- No outcome variable, just a set of predictors (features) measured on a set of samples.
- Objective is more fuzzy – find groups of samples that behave similarly, find features that behave similarly, find linear combinations of features with the most variation.
- Difficult to know how well you are doing.
- Different from supervised learning, but can be useful as a pre-processing step for supervised learning.

Machine learning and this course

Machine learning

- Machine learning refers to a vast set of tools for *understanding data*.
- For a quantitative response Y and a set of predictors X:

$$Y = f(X) + \epsilon$$

- Here, f represents the *systematic* information that X provides about Y.
- Statistical learning refers to a set of approaches for estimating f.
- Most of the course we'll spend talking about different ways to estimate f and how to evaluate whether we've done a good job with it.

Where does this course fit in?

- Supervised versus unsupervised learning.
- Regression versus classification.
- No single *best* method. We'll spend a lot of time choosing the most appropriate tool for a given dataset using different measures of the quality of fit. E.g. MSE

$$MSE_{training} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

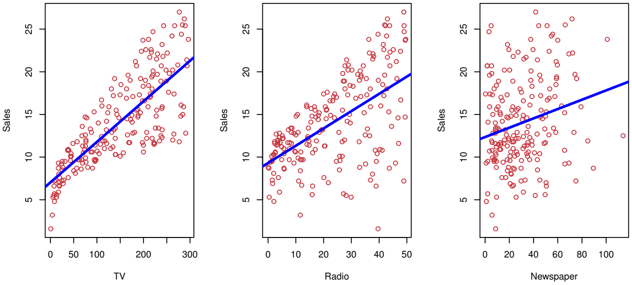
$$MSE_{test} = \mathrm{Ave}(\hat{f}(x_0) - y_0)^2$$

Why should we bother with f?

1. Prediction: $\widehat{Y} = \widehat{f}(X)$, where $\widehat{f}$ is a *black box*.
2. Inference: How Y is changing as a function of X.

- Depending on whether the ultimate goal is prediction, inference or a mix of both, we may deploy different methods for estimating f.
- Also depending on the ultimate goal you may or may not care about evaluating the causal relationship between Y and X.

What is Machine Learning?



- Shown are `Sales` vs `TV`, `Radio` and `Newspaper`, with a blue linear-regression line fit separately to each.
- We can predict `Sales` using a model

$$Sales \approx f(TV, Radio, Newspaper)$$

Notation

- Here `Sales` is a *response* or *target* that we wish to predict. We generically refer to the response as Y.
- `TV` is a *feature*, or *input*, or *predictor*; we name it X_1.
- Likewise name `Radio` as X_2, and so on.
- We can refer to the input vector collectively as

$$X = (X_1, X_2, X_3)$$

- Now we write our model as

$$Y = f(X) + \epsilon$$

where ϵ captures measurement errors and other discrepancies.

What is $f(X)$ good for?

- With a good f we can make predictions of Y at new points $X = x$.
- We can understand which components of $X = (X_1, X_2, \ldots, X_p)$ are important in explaining Y, and which are irrelevant. For example, `Seniority` and `Years of Education` have a big impact on `Income`, but `Marital Status` typically does not.
- Depending on the complexity of f, we may be able to understand how each component X_j of X affects Y.

- Is there an ideal $f(X)$? In particular, what is a good value for $f(X)$ at any selected value of X, say $X = 4$?
- There can be many Y values at $X = 4$. A good value is

$$f(4) = E(Y|X = 4)$$

- $E(Y|X = 4)$ means expected value (average) of Y given $X = 4$.
- This ideal $f(x) = E(Y|X = x)$ is called the regression function.

The regression function $f(x)$

- Is also defined for vector X; e.g. $f(x) = f(x_1, x_2, x_3) = E(Y|X_1 = x_1, X_2 = x_2, X_3 = x_3)$
- Is the ideal or optimal predictor of Y with regard to mean-squared prediction error: $f(x) = E(Y|X = x)$ is the function that minimizes $E[(Y - g(X))^2|X = x]$ over all functions g at all points $X = x$.
- $\epsilon = Y - f(x)$ is the irreducible error – i.e. even if we knew $f(x)$, we would still make errors in prediction, since at each $X = x$ there is typically a distribution of possible Y values.
- For any estimate $\hat{f}(x)$ of $f(x)$, we have

$$E[(Y - \hat{f}(X))^2|X = x] = \underbrace{[f(x) - \hat{f}(x)]^2}_{Reducible} + \underbrace{Var(\epsilon)}_{Irreducible}$$

How to estimate f

- Typically we have few if any data points with $X = 4$ exactly.
- So we cannot compute $E(Y|X = x)$!
- Relax the definition and let

$$\hat{f}(x) = Ave(Y|X \in N(x))$$

  where $N(x)$ is some neighborhood of x.

- Nearest neighbor averaging can be pretty good for small p – i.e. $p \leq 4$ and large-ish N.
- We will discuss smoother versions, such as kernel and spline smoothing later in the course.
- Nearest neighbor methods can be lousy when p is large. Reason: the curse of dimensionality. Nearest neighbors tend to be far away in high dimensions.
    - We need to get a reasonable fraction of the N values of y_i to average to bring the variance down – e.g. 10%.
    - A 10% neighborhood in high dimensions need no longer be local, so we lose the spirit of estimating $E(Y|X = x)$ by local averaging.

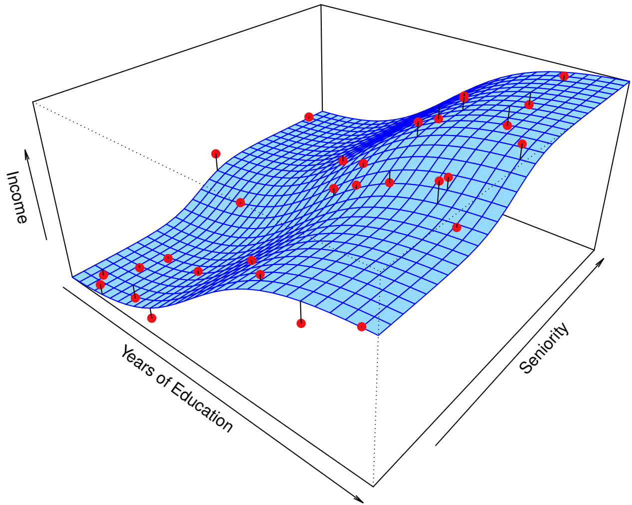
Parametric and structured models

The linear model is an important example of a parametric model:

$$f_L(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p.$$

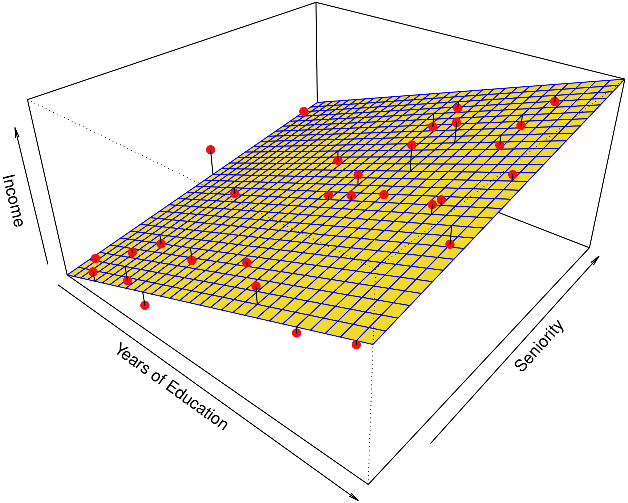
- A linear model is specified in terms of $p + 1$ parameters $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$.
- We estimate the parameters by fitting the model to training data.
- Although it is almost never correct, a linear model often serves as a good and interpretable approximation to the unknown true function $f(X)$.

Simulated example



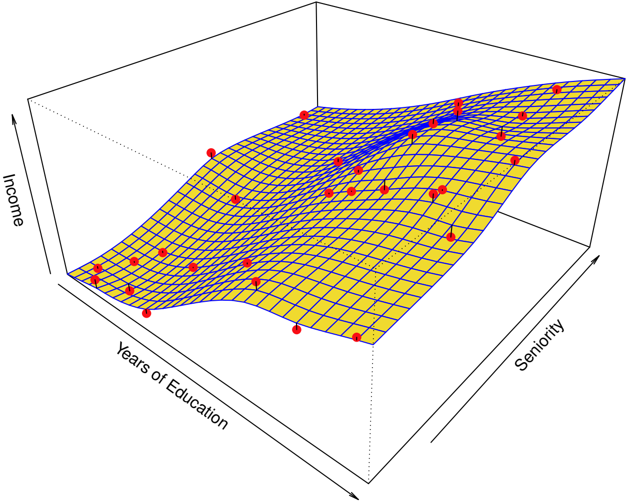
Red points are simulated values for `income` from the model

$$income = f(education, seniority) + \epsilon$$

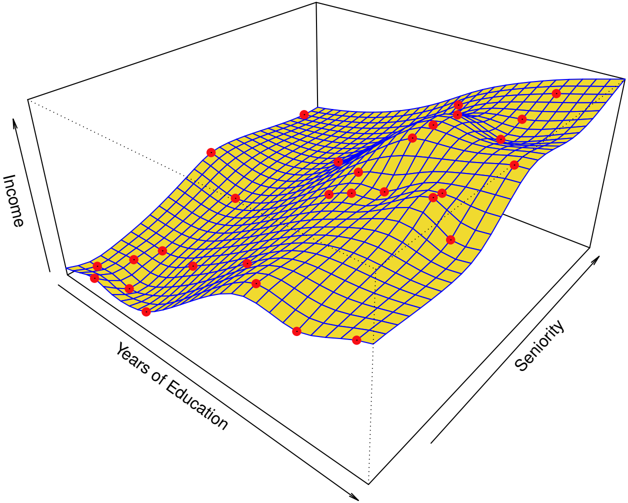


Linear regression model fit to the simulated data.

$$\hat{f}_L(education, seniority) = \hat{\beta}_0 + \hat{\beta}_1 \times education + \hat{\beta}_2 \times seniority$$



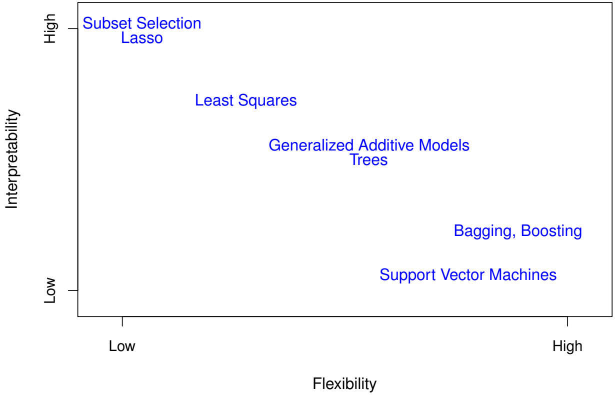
- More flexible regression model $\hat{f}_S(\textit{education}, \textit{seniority})$ fit to the simulated data.
- Here we use a technique called a thin-plate spline to fit a flexible surface.
- We control the roughness of the fit.



- Even more flexible spline regression model $\hat{f}_S(education, seniority)$ fit to the simulated data.
- Here the fitted model makes no errors on the training data!
- Also known as overfitting.

Some trade-offs

- Prediction accuracy versus interpretability.
    - Linear models are easy to interpret; thin-plate splines are not.
- Good fit versus over-fit or under-fit.
    - How do we know when the fit is just right?
- Parsimony versus black-box.
    - We often prefer a simpler model involving fewer variables over a black-box predictor involving them all.


Subset Selection
Lasso
Least Squares
Generalized Additive Models
Trees
Bagging, Boosting
Support Vector Machines
Interpretability
High
Low
Flexibility
Low
High

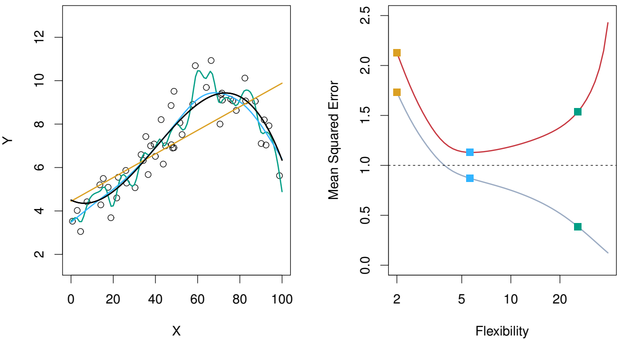
Assessing Model Accuracy

- Suppose we fit a model $\hat{f}(x)$ to some training data $Tr = \{x_i, y_i\}_1^N$, and we wish to see how well it performs.
- We could compute the average squared prediction error over Tr:

$$MSE_{Tr} = Ave_{i \in Tr}[y_i - \hat{f}(x_i)]^2$$

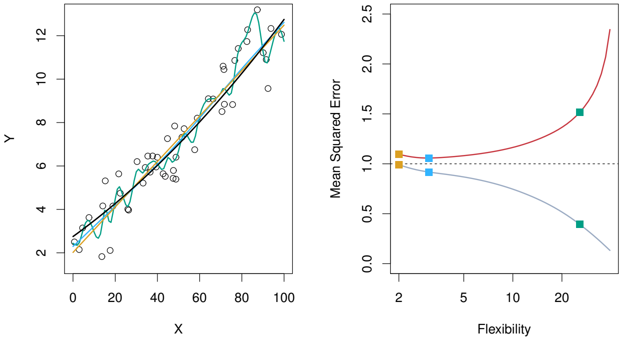
  This may be biased toward more overfit models.
- Instead we should, if possible, compute it using fresh test data $Te = \{x_i, y_i\}_1^M$:

$$MSE_{Te} = Ave_{i \in Te}[y_i - \hat{f}(x_i)]^2$$

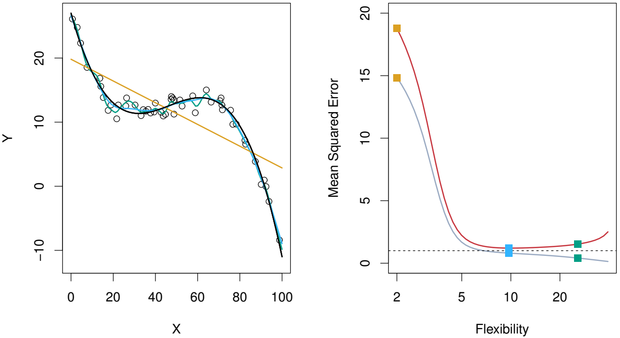


Data simulated from f, shown in black. Three estimates of f are shown: the linear regression line (orange curve), and two smoothing splines.

- Black curve is truth.
- Red curve on right is MSE_{Te}, grey curve is MSE_{Tr}.
- Orange, blue and green curves/squares correspond to fits of different flexibility.



- The setup as before, using a different true f that is much closer to linear. In this setting, linear regression provides a very good fit to the data.
- Here the truth is smoother, so the smoother fit and linear model do really well.



- Setup as above, using a different f that is far from linear.
- In this setting, linear regression provides a very poor fit to the data.
- Here the truth is wiggly and the noise is low, so the more flexible fits do the best.

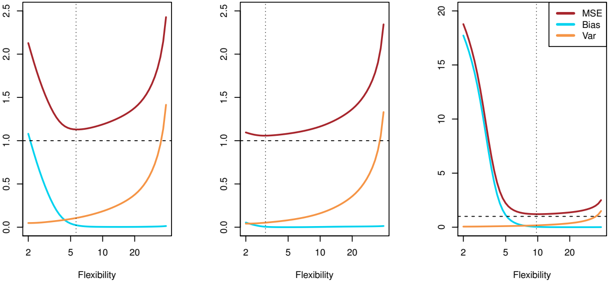
Bias-Variance Trade-off

- Suppose we have fit a model $f(x)$ to some training data Tr, and let (x_0, y_0) be a test observation drawn from the population.
- If the true model is $Y = f(X) + \epsilon$ (with $f(x) = E(Y|X = x)$), then

$$E(y_0 - \hat{f}(x_0)) = Var(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + Var(\epsilon).$$

- The expectation averages over the variability of y_0 as well as the variability in Tr. Note that $\mathrm{Bias}(\hat{f}(x_0)) = E[\hat{f}(x_0)] - f(x_0)$.
- Typically as the flexibility of $\hat{f}$ increases, its variance increases, and its bias decreases.
- So choosing the flexibility based on average test error amounts to a bias-variance trade-off.

Bias-variance trade-off for the three examples



Classification Problems

Here the response variable Y is qualitative – e.g. email is one of $\mathcal{C} = (spam, ham)$ ($ham = godemail$), digitclass is one of $\mathcal{C} = \{0, 1, \ldots, 9\}$. Our goals are to:

- Build a classifier $\mathcal{C}(X)$ that assigns a class label from $\mathcal{C}$ to a future unlabeled observation X.
- Assess the uncertainty in each classification.
- Understand the roles of the different predictors among $X = (X_1, X_2, \ldots, X_p)$.

- Is there an ideal $\mathcal{C}(X)$? Suppose the K elements in $\mathcal{C}$ are numbered $1, 2, \ldots, K$. Let

$$p_k(x) = Pr(Y = k|X = x), k = 1, 2, \ldots, K.$$

- These are the conditional class probabilities at x. Then the Bayes optimal classifier at x is

$$\mathcal{C}(x) = j \text{ if } p_j(x) = \max\{p_1(x), p_2(x), \ldots, p_K(x)\}$$

- Nearest-neighbor averaging can be used as before.
- Also breaks down as dimension grows.
- However, the impact on $\hat{\mathcal{C}}(x)$ is less than on $\hat{p}_k(x)$, $k = 1, \ldots, K$.

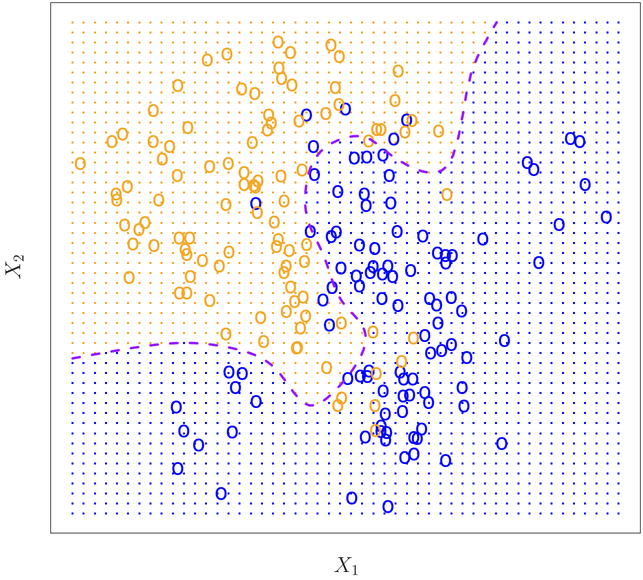
Classification: some details

- Typically we measure the performance of $\hat{\mathcal{C}}(x)$ using the misclassification error rate:

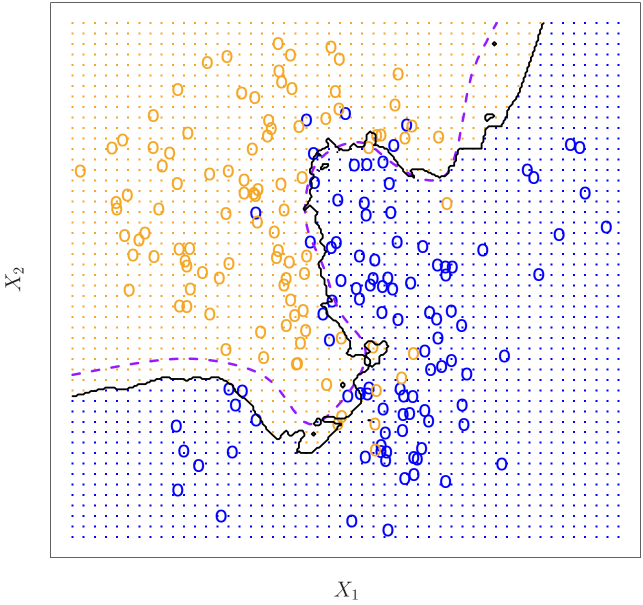
$$Err_{Te} = Ave_{i \in Te} \mathcal{I}[y_i \neq \hat{\mathcal{C}}(x_i)]$$

- The Bayes classifier (using the true $p_k(x)$) has smallest error (in the population).
- Support-vector machines build structured models for $\mathcal{C}(x)$.
- We will also build structured models for representing the $p_k(x)$. For example, logistic regression, generalized additive models.

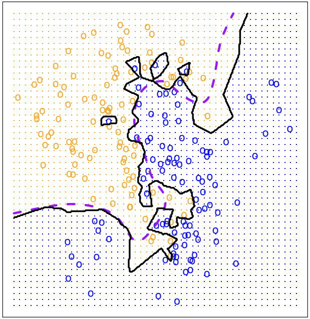
Example: K-nearest neighbors in two dimensions

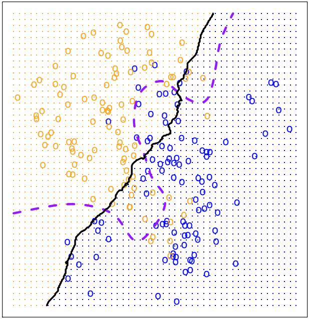


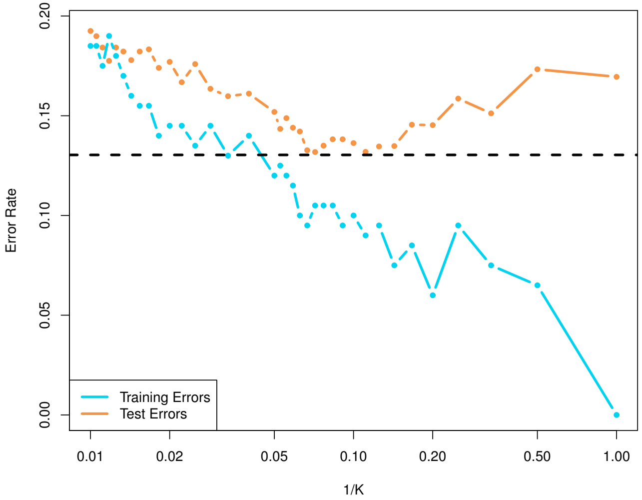
KNN: K=10



KNN: K=1 **KNN: K=100**







Error Rate
0.00
0.05
0.10
0.15
0.20
Training Errors
Test Errors
0.01
0.02
0.05
0.10
0.20
0.50
1.00
1/K